Talking Robots With LEGO MindStorms

Alexander Koller
Saarland University
Saarbrücken, Germany
koller@coli.uni-sb.de

Geert-Jan M. Kruijff
Saarland University
Saarbrücken, Germany
gj@coli.uni-sb.de

Abstract

This paper shows how talking robots can be built from off-the-shelf components, based on the Lego MindStorms robotics platform. We present four robots that students created as final projects in a seminar we supervised. Because Lego robots are so affordable, we argue that it is now feasible for any dialogue researcher to tackle the interesting challenges at the robot-dialogue interface.

1 Introduction

Ever since Karel Čapek introduced the word "robot" in his 1921 novel *Rossum's Universal Robots* and the subsequent popularisation through Issac Asimov's books, the idea of building autonomous robots has captured people's imagination. The creation of an intelligent, talking robot has been the ultimate dream of Artificial Intelligence from the very start.

Yet, although there has been a tremendous amount of AI research on topics such as control and navigation for robots, the issue of integrating dialogue capabilities into a robot has only recently started to receive attention. Early successes were booked with Flakey (Konolige et al., 1993), a voice-controlled robot which roamed the corridors of SRI. Since then, the field of *socially interactive robots* has established itself (see (Fong et al., 2003)). Often-cited examples of such interactive robots that have a capability of communicating in natural language are the humanoid robot ROBOVIE (Kanda et al., 2002) and robotic museum tour guides like RHINO (Burgard et al., 1999) (Deutsches Museum Bonn), its successor MINERVA touring the Smithsonian in Washington (Thrun et al., 2000), and ROBOX at the Swiss National Exhibition Expo02 (Siegwart and et al, 2003). However, dialogue systems used in robotics appear to be mostly restricted to relatively simple finite-state, query/response interaction. The only robots involving dialogue systems that are state-of-the-art in computational linguistics (and that we are aware of)

are those presented by Lemon et al. (2001), Sidner et al. (2003) and Bos et al. (2003), who equipped a mobile robot with an information state based dialogue system.

There are two obvious reasons for this gap between research on dialogue systems in robotics on the one hand, and computational linguistics on the other hand. One is that the sheer cost involved in buying or building a robot makes traditional robotics research available to only a handful of research sites. Another is that building a talking robot combines the challenges presented by robotics and natural language processing, which are further exacerbated by the interactions of the two sides.

In this paper, we address at least the first problem by demonstrating how to build talking robots from affordable, commercial off-the-shelf (COTS) components. We present an approach, tested in a seminar taught at the Saarland University in Winter 2002/2003, in which we combine the Lego MindStorms system with COTS software for speech recognition/synthesis and dialogue modeling.

The Lego MindStorms[1] system extends the traditional Lego bricks with a central control unit (the RCX), as well as motors and various kinds of sensors. It provides a severely limited computational platform from a traditional robotics point of view, but comes at a price of a few hundred, rather than tens of thousands of Euros per kit. Because MindStorms robots can be flexibly connected to a dialogue system running on a PC, this means that affordable robots are now available to dialogue researchers.

We present four systems that were built by teams of three students each under our supervision, and use off-the-shelf components such as the MindStorms kits, a dialogue system, and a speech recogniser and synthesis system, in addition to communications software that we ourselves wrote to link all the components together. It turns out that using

---

[1]LEGO and LEGO MindStorms are trademarks of the LEGO Company.

this accessible technology, it is possible to create basic but interesting talking robots in limited time (7 weeks). This is relevant not only for future research, but can also serve as a teaching device that has shown to be extremely motivating for the students. MindStorms are a staple in robotics education (Yu, 2003; Gerovich et al., 2003; Lund, 1999), but to our knowledge, they have never been used as part of a language technology curriculum.

The paper is structured as follows. We first present the basic setup of the MindStorms system and the software architecture. Then we present the four talking robots built by our students in some detail. Finally, we discuss the most important challenges that had to be overcome in building them. We conclude by speculating on further work in Section 5.

2 Architecture

Lego MindStorms robots are built around a programmable microcontroller, the RCX. This unit, which looks like an oversized yellow Lego brick, has three ports each to attach sensors and motors, an infrared sender/receiver for communication with the PC, and 32 KB memory to store the operating system, a programme, and data.
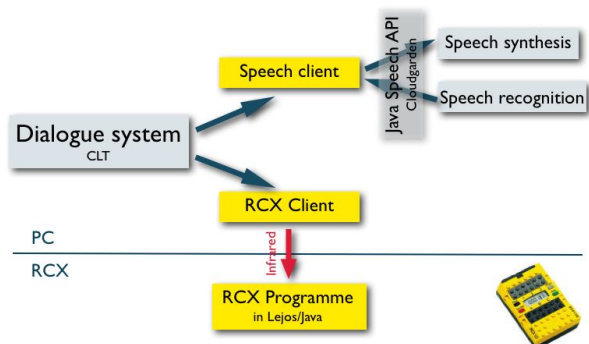

Figure 1: Architecture of a talking Lego robot.

Our architecture for talking robots (Fig. 1) consists of four main modules: a dialogue system, a speech client with speech recognition and synthesis capabilities, a module for infrared communication between the PC and the RCX, and the programme that runs on the RCX itself. Each student team had to specify a dialogue, a speech recognition grammar, and the messages exchanged between PC and RCX, as well as the RCX control programme. All other components were off-the-shelf systems that were combined into a larger system by us.

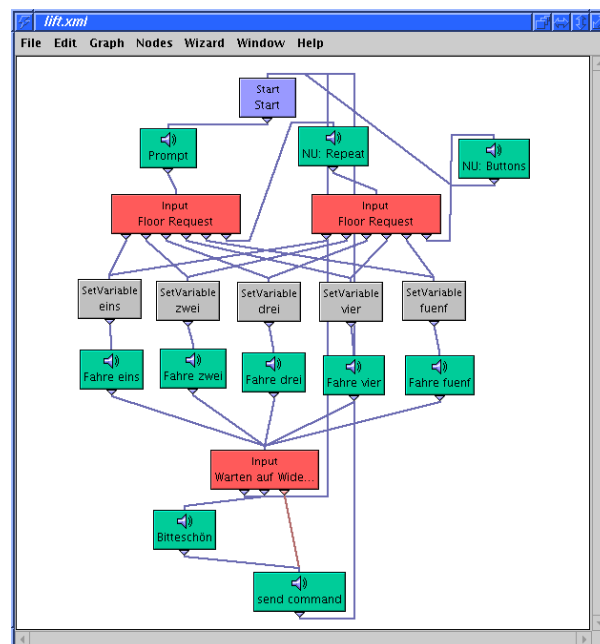The centrepiece of the setup is the dialogue system. We used the DiaWiz system by CLT

Figure 2: The dialogue system.

Sprachtechnologie GmbH[2], a proprietary framework for defining finite-state dialogues (McTear, 2002). It has a graphical interface (Fig. 2) that allows the user to draw the dialogue states (shown as rectangles in the picture) and connect them via edges. The dialogue system connects to an arbitrary number of "clients" via sockets. It can send messages to and receive messages from clients in each dialogue state, and thus handles the entire dialogue management. While it was particularly convenient for us to use the CLT system, it could probably replaced without much effort by a VoiceXML-based dialogue manager.

The client that interacts most directly with the user is a module for speech recognition and synthesis. It parses spoken input by means of a recognition grammar written in the Java Speech Grammar Format, [3] and sends an extremely shallow semantic representation of the best recognition result to the dialogue manager as a feature structure. The output side can be configured to either use a speech synthesiser, or play back recorded WAV files. Our implementation assumes only that the recognition and synthesis engines are compliant with the Java Speech API [4].

The IR communication module has the task of converting between high-level messages that the di-

---

[2]http://www.clt-st.de

[3]http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/
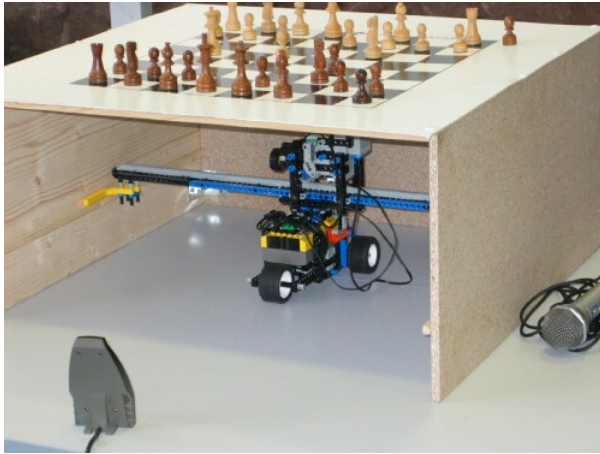
[4]http://java.sun.com/products/java-media/speech/

Figure 3: A robot playing chess.

alogue manager and the RCX programme exchange and their low-level representations that are actually sent over the IR link, in such a way that the user need not think about the particular low-level details. The RCX programme itself is again implemented in Java, using the Lejos system (Bagnall, 2002). Such a programme is typically small (to fit into the memory of the microcontroller), and reacts concurrently to events such as changes in sensor values and messages received over the infrared link, mostly by controlling the motors and sending messages back to the PC.

3 Some Robots

3.1 Playing Chess

The first talking robot we present plays chess against the user (Fig. 3). It moves chess pieces on a board by means of a magnetic arm, which it can move up and down in order to grab and release a piece, and can place the arm under a certain position by driving back and forth on wheels, and to the right and left on a gear rod.

The dialogue between the human player and the robot is centred around the chess game: The human speaks the move he wants to make, and the robot confirms the intended move, and announces check and checkmate. In order to perform the moves for the robot, the dialogue manager connects to a specialised client which encapsulates the GNU Chess system.[5] In addition to computing the moves that the robot will perform, the chess programme is also used in disambiguating elliptical player inputs.

Figure 4 shows the part of the chess dialogue model that accepts a move as a spoken command from the player. The Input node near the top waits for the speech recognition client to report that it
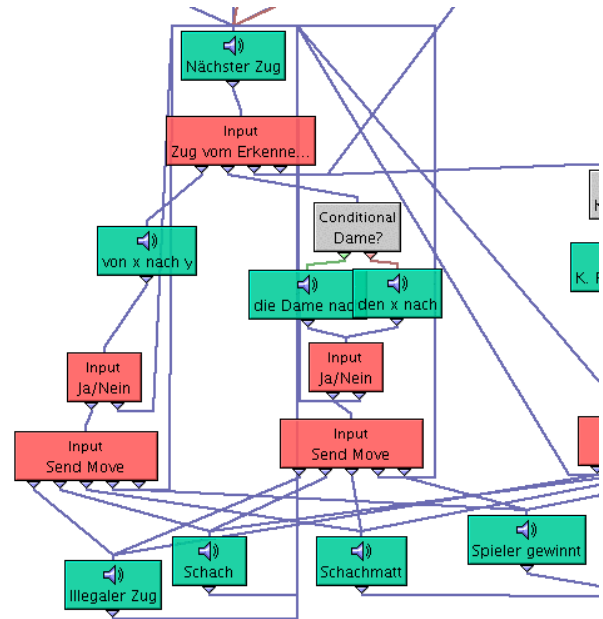
---

[5]http://www.gnu.org/software/chess/chess.html

Figure 4: A small part of the Chess dialogue.

```
<cmd> = [<move>] <piece> <to> <squareTo>
        | ...
<squareTo> = <colTo> <rowTo>
<colTo> = [a wie] anton {colTo:a} |
          [b wie] berta {colTo:b} | ...
<rowTo> = eins {rowTo:1} |
          zwei {rowTo:2} | ...
```

Figure 5: A small part of the Chess grammar.

understood a player utterance as a command. An excerpt from the recogniser grammar is shown in Fig. 5: The grammar is a context-free grammar in JSGF format, whose production rules are annotated with *tags* (in curly brackets) representing a very shallow semantics. The tags for all production rules used in a parse tree are collected into a table.

The dialogue manager then branches depending on the type of the command given by the user. If the command specified the piece and target square, e.g. "move the pawn to e4", the recogniser will return a representation like {piece="pawn" colTo="e" rowTo="4"}, and the dialogue will continue in the centre branch. The user can also specify the source and target square.

If the player confirms that the move command was recognised correctly, the manager sends the move description to the chess client (the "send move" input nodes near the bottom), which can disambiguate the move description if necessary, e.g. by expanding moves of type "move the pawn to

e4" to moves of type "move from e2 to e4". Note that the reference "the pawn" may not be globally unique, but if there is only one possible referent that could perform the requested move, the chess client resolves this automatically.

The client then sends a message to the RCX, which moves the piece using the robot arm. It updates its internal data structures, as well as the GNU Chess representations, computes a move for itself, and sends this move as another message to the RCX.

While the dialogue system as it stands already offers some degree of flexibility with regard to move phrasings, there is still plenty of open room for improvements. One is to use even more context information, in order to understand commands like "take it with the rook". Another is to incorporate recent work on improving recognition results in the chess domain by certain plausibility inferences (Gabsdil, 2004).

3.2 Playing a Shell Game

Figure 6 introduces *Luigi Legonelli*. The robot represents a charismatic Italian shell-game player, and engages a human player in style: Luigi speaks German with a heavy Italian accent, lets the human player win the first round, and then tries to pull several tricks either to cheat or to keep the player interested in the game.

Figure 6: A robot playing a shell game.

Luigi's Italian accent was obtained by feeding transliterated German sentences to a speech synthe-sizer with an Italian voice. Although the resulting accent sounded authentic, listeners who were unfamiliar with the accent had trouble understanding it. For demonstration purposes we therefore decided to use recorded speech instead. To this end, the Italian student on the team lent his voice for the different sentences uttered by Luigi.

The core of Luigi's dialogue model reflects the progress of game play in a shell game. At the start, Luigi and the player settle on a bet (between 1 and 10 euros), and Luigi shows under which shell the coin is. Then, Luigi manipulates the shells (see also below), moving them (and the coin) around the board, and finally asks the player under which shell the player believes the coin is. Upon the player's guess Luigi lifts the shell indicated by the player, and either loudly exclaims the unfairness of life (if he has lost) or kindly inquires after the player's visual capacities (in case the player has guessed wrong). At the end of the turn, Luigi asks the player whether he wants to play again. If the player would like to stop, Luigi tries to persuade the player to stay; only if the player is persistent, Luigi will end the game and beat a hasty retreat.

(1) **rob** "Ciao, my name is Luigi Legonelli. Do you feel like a little game?"

    usr "Yes ... "

    rob "The rules are easy. I move da cuppa, you know, cuppa? You look, say where coin is. How much money you bet?"

    usr "10 Euros."

    rob (Luigi moves the cups/shells)

    rob "So, where is the coin? What do you think, where's the coin?"

    usr "Cup 1"

    rob "Mamma mia! You have won! Who told you, where is coin?! Another game? Another game!"

    usr "No."

    rob "Come! Play another game!"

    usr "No."

    rob "Okay, ciao signorina! Police, much police! Bye bye!"

The shells used in the game are small cups with a metal top (a nail), which enables Luigi to pick them up using a "hand" constructed around a magnet. The magnet has a downward oriented, U-shaped construction that enables Luigi to pick up two cups at the same time. Cups then get moved around the board by rotating the magnet. By magnetizing the nail at the top of the cup, not only the cup but

also the coin (touched by the tip of the nail) can be moved. When asked to show whether the coin is under a particular shell, one of Luigi's tricks is to keep the nail magnetized when lifting a cup – thus also lifting the coin, giving off the impression that there was no coin under the shell.

The Italian accent, the android shape of the robot, and the 'authentic' behavior of Luigi all contributed to players genuinely getting engaged in the game. After the first turn, having won, most players acknowledged that this is an amusing Lego construction; when they were tricked at the end of the second turn, they expressed disbelief; and when we showed them that Luigi had deliberately cheated them, astonishment. At that point, Luigi had ceased to be simply an amusing Lego construction and had achieved its goal as an entertainment robot that can immerse people into its game.

3.3 Exploring a pyramid

The robot in Figure 7, dubbed "Indy", is inspired by the various robots that have been used to explore the Great Pyramids in Egypt (e.g. Pyramid Rover[6], UPUAUT[7]). It has a digital videocamera (webcam) and a lamp mounted on it, and continually transmits images from inside the pyramid. The user, watching the images of the videocamera on a computer screen, can control the robot's movements and the angle of the camera by voice.

Figure 7: A robot exploring a pyramid.

Human-robot interaction is crucial to the exploration task, as neither user nor robot has a complete picture of the environment. The robot is aware of the environment through an (all-round) array of touch-sensors, enabling it to detect e.g. openings in walls; the user receives a more detailed picture, but

[6]http://www.newscientist.com/news/news.jsp?id=ns99992805
[7]http://www.cheops.org

only of the environment straight ahead of the robot (due to the frontal orientation of the camera).

The dialogue model for Indy defines the possible interaction that enables Indy and the user to jointly explore the environment. The user can initiate a dialogue to control the camera and its orientation (by letting the robot turn on the spot, in a particular direction), or to instruct the robot to make particular movements (i.e. turn left or right, stop).

3.4 Traversing a labyrinth

A variation on the theme of human-robot interaction in navigation is the robot in Figure 8. Here, the user needs to guide a robot through a labyrinth, specified by thick black lines on a white background. The task that the robot and the human must solve collaboratively is to pick up objects randomly strewn about the maze. The robot is able to follow the black lines lines (the "path") by means of an array of three light sensors at its front.
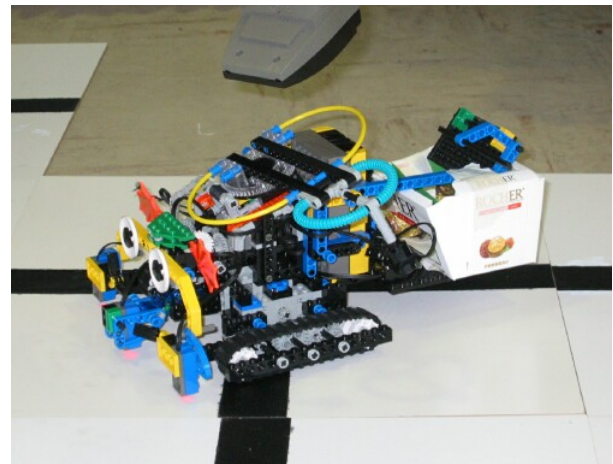

Figure 8: A robot traversing a labyrinth.

Both the user and the robot can take the initiative in the dialogue. The robot, capable of spotting crossings (and the possibilities to go straight, left and/or right), can initiate a dialogue asking for directions if the user had not instructed the robot beforehand; see Example 2.

(2) **rob** (The robot arrives at a crossing; it recognizes the possibility to go either straight or left; there are no current instructions)

    rob "I can go left or straight ahead; which way should I go?"

    usr "Please go right."

    rob "I cannot go right here.

    usr "Please go straight."

    rob "Okay."

The user can give the robot two different types of directions: *in-situ* directions (as illustrated in Example 2) or *deictic* directions (see Example 3 below). This differentiates the labyrinth robot from the pyramid robot described in §3.3, as the latter could only handle in-situ directions.

(3) **usr** "Please turn left at the next crossing."

   rob "Okay"

   rob (The robot arrives at a crossing; it recognizes the possibility to go either straight or left; it was told to go left at the next crossing)

   rob (The robot recognizes it can go left and does so, as instructed)

4   Discussion

The first lesson we can learn from the work described above is that affordable COTS products in dialogue and robotics have advanced to the point that it is feasible to build simple but interesting talking robots with limited effort. The Lego Mind-Storms platform, combined with the Lejos system, turned out to be a flexible and affordable robotics framework. More "professional" robots have the distinct advantage of more interesting sensors and more powerful on-board computing equipment, and are generally more physically robust, but Lego MindStorms is more than suitable for robotics experimentation under controlled circumstances.

Each of the robots was designed, built, and programmed within twenty person-weeks, after an initial work phase in which we created the basic infrastructure shown in Figure 1. One prerequisite of this rather efficient development process was that the entire software was built on the Java platform, and was kept highly modular. Speech software adhering to the Java Speech API is becoming available, and plugging e.g. a different JSAPI-compliant speech recogniser into our system is now a matter of changing a line in a configuration file.

However, building talking robots is still a challenge that combines the particular problems of dialogue systems and robotics, both of which introduce situations of incomplete information. The dialogue side has to robustly cope with speech recognition errors, and our setup inherits all limitations inherent in finite-state dialogue; applications having to do e.g. with information seeking dialogue would be better served with a more complex dialogue model. On the other hand, a robot lives in the real world, and has to deal with imprecisions in measuring its position, unexpected obstacles, communications with the PC breaking off, and extremely limited sensory information about its surroundings.

5   Conclusion

The robots we developed together with our students were toy robots, looked like toy robots, and could (given the limited resources) only deal with toy examples. However, they confirmed that there are affordable COTS components on the market with which we can, even in a limited amount of time, build engaging talking robots that capture the essence of various (potential) real-life applications. The chess and shell game players could be used as entertainment robots. The labyrinth and pyramid robots could be extended into tackling real-world exploration or rescue tasks, in which robots search for disaster victims in environments that are too dangerous for rescuers to venture into.[8] Dialogue capabilities are useful in such applications not just to communicate with the human operator, but also possibly with disaster victims, to check their condition.

Moreover, despite the small scale of these robots, they show genuine issues that could provide interesting lines of research at the interface between robotics and computational linguistics, and in computational linguistics as such. Each of our robots could be improved dramatically on the dialogue side in many ways. As we have demonstrated that the equipment for building talking robots is affordable today, we invite all dialogue researchers to join us in making such improvements, and in investigating the specific challenges that the combination of robotics and dialogue bring about. For instance, a robot moves and acts in the real world (rather than a carefully controlled computer system), and suffers from uncertainty about its surroundings. This limits the ways in which the dialogue designer can use visual context information to help with reference resolution.

Robots, being *embodied* agents, present a host of new challenges beyond the challenges we face in computational linguistics. The interpretation of language needs to be *grounded* in a way that is both based in perception, and on conceptual structures to allow for generalization over experiences. Naturally, this problem extends to the acquisition of language, where approaches such as (Nicolescu and Matarić, 2001; Carbonetto and Freitos, 2003; Oates, 2003) have focused on basing understanding entirely in sensory data.

Another interesting issue concerns the interpretation of deictic references. Research in multi-modal

---

[8]See also http://www.rescuesystem.org/robocuprescue/

interfaces has addressed the issue of deictic reference, notably in systems that allow for pen-input (see (Oviatt, 2001)). Embodied agents raise the complexity of the issues by offering a broader range of sensory input that needs to be combined (cross-modally) in order to establish possible referents.

References

Brian Bagnall. 2002. *Core Lego Mindstorms Programming*. Prentice Hall, Upper Saddle River NJ.

Johan Bos, Ewan Klein, and Tetsushi Oka. 2003. Meaningful conversation with a mobile robot. In *Proceedings of the 10th EACL*, Budapest.

W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. 1999. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2):3–55.

Peter Carbonetto and Nando de Freitos. 2003. Why can't José talk? the problem of learning semantic associations in a robot environment. In *Proceedings of the HLT-NAACL 2003 Workshop on Learning Word Meaning from Non-Linguistic Data*, pages 54–61, Edmonton, Canada.

Terrence W Fong, Illah Nourbakhsh, and Kerstin Dautenhahn. 2003. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42:143–166.

Malte Gabsdil. 2004. Combining acoustic confidences and pragmatic plausibility for classifying spoken chess move instructions. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*.

Oleg Gerovich, Randal P. Goldberg, and Ian D. Donn. 2003. From science projects to the engineering bench. *IEEE Robotics & Automation Magazine*, 10(3):9–12.

Takayuki Kanda, Hiroshi Ishiguro, Tetsuo Ono, Michita Imai, and Ryohei Nakatsu. 2002. Development and evaluation of an interactive humanoid robot "robovie". In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2002)*, pages 1848–1855.

Kurt Konolige, Karen Myers, Enrique Ruspini, and Alessandro Saffiotti. 1993. Flakey in action: The 1992 aaai robot competition. Technical Report 528, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Apr.

Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters. 2001. A multi-modal dialogue system for human-robot conversation. In *Proceedings NAACL 2001*.

Henrik Hautop Lund. 1999. AI in children's play with LEGO robots. In *Proceedings of AAAI 1999 Spring Symposium Series*, Menlo Park. AAAI Press.

Michael McTear. 2002. Spoken dialogue technology: enabling the conversational user interface. *ACM Computing Surveys*, 34(1):90–169.

Monica N. Nicolescu and Maja J. Matarić. 2001. Learning and interacting in human-robot domains. *IEEE Transactions on Systems, Man and Cybernetics*, 31.

Tim Oates. 2003. Grounding word meanings in sensor data: Dealing with referential uncertainty. In *Proceedings of the HLT-NAACL 2003 Workshop on Learning Word Meaning from Non-Linguistic Data*, pages 62–69, Edmonton, Canada.

Sharon L. Oviatt. 2001. Advances in the robust processing of multimodal speech and pen systems. In P. C. Yuen, Y.Y. Tang, and P.S. Wang, editors, *Multimodal InterfacesB for Human Machine Communication*, Series on Machine Perception and Artificial Intelligence, pages 203–218. World Scientific Publisher, London, United Kingdom.

Candace L. Sidner, Christopher Lee, and Neal Lesh. 2003. Engagement by looking: Behaviors for robots when collaborating with people. In *Proceedings of the 7th workshop on the semantics and pragmatics of dialogue (DIABRUCK)*.

R. Siegwart and et al. 2003. Robox at expo.02: A large scale installation of personal robots. *Robotics and Autonomous Systems*, 42:203–222.

S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. 2000. Probabilistic algorithms and the interactive museum tour-guide robot minerva. *International Journal of Robotics Research*, 19(11):972–999.

Xudong Yu. 2003. Robotics in education: New platforms and environments. *IEEE Robotics & Automation Magazine*, 10(3):3.